Comment faire un ODD magiquement

Lou Burnard *2017-02-18*

1 Introduction

On n'a de cesse de vous dire de faire un ODD pour votre projet. Mais il y existe sûrement une manière de démarrer rapidement le processus automatiquement ? Eh oui, c'est possible ! Ce guide vous présente comment générer un ODD automatiquement à partir d'un corpus existant de documents TEI P5, et vous propose quelques suggestions pour pouvoir l'améliorer.

2 Installer la transformation ODD by Example

La Text Encoding Initiative fournit un utilitaire dénommé ODD By Example qui est capable de traiter un ensemble de documents TEI existants et d'en extraire une description des éléments et des classes TEI qu'ils emploient. Cet utilitaire est une feuille de style XSLT qui fait maintenant partie du paquet standard TEI Stylesheet. Vous trouverez le fichier tools/oddbyexample.xsl, dans le répertoire que vous avez utilisé pour installer le paquet des Stylesheet ; sur un système Linux typique, ce répertoire sera nommé /usr/share/xml/tei/stylesheet. Nous n'expliquons pas ici comment installer ce paquet.

Cette transformation est légèrement différente des feuilles de style XSLT habituelle du fait qu'elle est conçue pour traiter un grand nombre de documents séparés plutôt qu'un simple document. Vous pouvez contrôler ses entrées et ses sorties au moyen de paramètres qui doivent être passés au processeur XSLT de manière légèrement distincte en fonction de votre environnement. Par exemple, vous pouvez exécuter la transformation en ligne de commande, en utilisant une commande comme saxon directement, ou bien vous pouvez la lancer au sein du logiciel oXygen en ayant d'abord configuré un *scénario de transformation* approprié.

Si vous êtes habitué à travailler en ligne de commande, cela pourrait être l'option la plus simple et la plus rapide. Pour traiter tous les fichiers du répertoire suivant /home/me/myTEICorpus et produire un fichier ODD d'exemple appelé myGenerated.odd dans le répertoire courant, vous pourriez entrer une commande comme celle-ci : `saxon -it:main -o:myGenerated.odd /usr/share/xml/tei/stylesheet/tools/oddbyexample.xsl corpus=/home/me/myCorpus`

(Si vous vous posiez la question, l'option `-it` indique à saxon quelle règle modèle (template) de la feuille de style doit être traitée en premier.)

Pour définir un scénario de transformation approprié dans oXygen pour les mêmes fichiers, vous pourriez procéder comme suit :

- Dans oXygen, ouvrir un des fichiers du répertoire /home/me/myTEICorpus
- Choisir Transformation -> Configurer le/les scénario(s) de transformation depuis le menu Document
- Cliquer sur Nouveau et choisissez "XML Transformation with XSLT"
- Donner un nom à votre scénario (par exemple, oddByExample)
- Laisser XML URL tel quel. Changer XSL URL pour pointer vers la feuille de style XSLT oddbyexample.xsl dans le répertoire de votre Framework TEI. Entrer `${frameworks}/tei/xml/tei/stylesheet/tools/oddbyexample.xsl` pour le trouver (oui, il *y a bien* deux `tei` dans ce chemin)
- Choisir Saxon-PE 9.4.0.4 comme processeur

- Cliquer sur la petite roue à engrenage près de cette boîte de dialogue pour choisir les Options avancées : vous avez besoin de renseigner la valeur `main` pour Template("-it") puis valider
- Cliquer ensuite sur le bouton Paramètres : vous devez régler le paramètre `corpus` pour contenir le nom complet du répertoire que vous voulez analyser. Supposant que vous avez ouvert un des fichiers mentionnés à la première étape ci-dessus, vous pouvez seulement donner au paramètre la valeur `${cfd}` et cliquer sur OK.
- Maintenant, sélectionner l'onglet Sortie…
  - Dans Enregistrer sous fournir un nom de fichier pour la sortie comme generated.odd
  - Cocher la case Ouvrir dans l'éditeur
  - Sélectionner le bouton radio XML dans Afficher la vue des résultats comme et cliquer sur OK
- Lancer la transformation en cliquant sur le bouton Appliquer les scénarios associés

Une fois que vous avez défini ce scénario de transformation, vous pouvez y avoir recours aussi souvent que vous voulez avec n'importe quelle collection de fichiers. Vous n'avez pas besoin de suivre à nouveau l'ensemble des étapes précédentes à chaque fois ! La prochaine fois, procédez comme suit :

- Dans oXygen, ouvrir n'importe quel fichier TEI-XML de la collection que vous voulez traiter
- Choisir Transformation -> Configurer le/les scénario(s) de transformation depuis le menu Document comme précédemment
- Vous pouvez maintenant voir le scénario que vous venez de définir dans la liste des scénarios "Globaux". Cocher la case à côté et cliquer sur le bouton Appliquer le/les scénario(s) associés

Une fois que cette association est faite, chaque fois que vous ouvrez un fichier TEI-XML dans oXygen, vous pouvez lancer à nouveau la transformation en cliquant simplement sur le bouton avec un grand triangle rouge dans la barre d'outils (ou bien en pressant simultanément les touches CTRL-SHIFT-T, ou bien en sélectionnant Document -> Transformation -> Appliquer le/les scénario de transformation)

Vous pouvez également éditer le scénario, par exemple en changeant l'une des valeurs de paramètres passés à la transformation, ou en changeant les options de sorties. Vous trouverez ci-dessous une liste des paramètres que vous pouvez modifier.

3 Comprendre la sortie

L'ODD généré par oddbyexample peut, à première vue, paraître un peu étrange. Parcourir le fichier pour trouver les éléments `<moduleRef>` car ceux-ci sont particulièrement utiles. Vous devriez voir les lignes suivantes

```
<moduleRef key="namesdates"
 include="persName listPerson person"/>
```

Cela nous indique que notre corpus utilise les éléments `<persName>`, `<listPerson>` et `<person>` qui sont fournis par le module TEI namesdates. Aucun autre élément de ce module n'est utilisé dans ce corpus.

Toutefois, mon ODD contient aussi de nombreuses lignes moins utiles. Par exemple, si un des attribut globaux (comme *@corresp*) a été utilisé sur seulement un élément dans notre corpus, l'ODD généré doit le supprimer explicitement de tous les éléments sur lesquels il n'est *pas* employé. Plus ennuyeux encore, peut-être, l'ODD généré contient toujours les déclarations de toutes les classes spécifiques aux modules que les éléments dudit module aient été utilisés ou non. Les déclarations n'ont aucun effet sur le schéma qui est généré, mais ils rendent plus compliqué à comprendre ce que fait l'ODD proprement dit.

Plus utile, l'ODD généré peut vous indiquer les valeurs qui sont effectivement employées pour les attributs (par exemple pour les valeurs de *@type*) qui ont le type de données (datatype) 'teidata.enumerated'.

Voici un exemple de ce que nous voulons dire :

```
<elementSpec ident="persName" mode="change">
 <attList>
  <attDef ident="corresp" mode="delete"/>
  <attDef ident="when" mode="delete"/>
  <attDef ident="notBefore" mode="delete"/>
  <attDef ident="notAfter" mode="delete"/>
  <attDef ident="key" mode="delete"/>
  <attDef ident="type" mode="change">
   <valList mode="add" type="closed">
    <valItem ident="actor"/>
   </valList>
  </attDef>
  <attDef ident="subtype" mode="delete"/>
 </attList>
</elementSpec>
```

Cette spécification nous indique que dans notre corpus la seule valeur réellement spécifiée pour l'attribut *@type* sur l'élément <persName> est 'actor'. Cela m'indique également que la plupart des autres attributs disponibles pour les classes dont <persName> est membre ne sont jamais effectivement employés. Par exemple, les attributs *@when*, *@notBefore* et *@notAfter* sont tous fournis par la classe att.datable, de laquelle <persName> est membre. Ces attributs ne sont utilisés sur aucune instance de <persName> au sein de notre corpus. Ils sont cependant utilisés sur les instances d'au moins un autre membre de cette classe (en fait <date>), ainsi la classe d'attribut elle-même ne peut pas être supprimée de l'ODD. Au lieu de cela, comme nous allons le voir, ils doivent être explicitement effacées de chaque éléments qui ne les utilise pas.

4 Utiliser et améliorer un ODD généré

Que pouvons-nous faire avec ce fichier ODD, à part l'étudier comme témoin des folies de nos encodeurs ? Nous pouvons, bien sûr, l'utiliser pour générer un schéma ou une documentation de nos pratiques, comme on l'aurait fait avec n'importe quel autre ODD. Voici un rappel de la manière dont vous pouvez configurer un scénario de transformation avec oXygen pour cela :

- Ouvrir le fichier ODD généré dans oXygen
- Dans le menu Document, choisir Transformation -> Configurer le/les scénario(s) de transformation, ou presser les touches CTRL-SHIFT-C, ou bien cliquer sur l'icône représentant une clef-à-molette dans la barre d'outil.
- Une liste des scénarios de transformation disponibles est présentée. Cocher sur la case adjacente au scénario que vous souhaitez appliquer : probablement TEI ODD XHTML et TEI ODD to RELAXNG Compact
- Cliquer le bouton Appliquer le/les scénario(s) associés pour lancer la ou les transformations ; vous pouvez ignorer les messages qui s'affichent dans la fenêtre du bas

- Si tout se déroule comme il faut, le XHTML va s'afficher dans votre navigateur par défaut, tandis que la sortie RelaxNG compacte sera sauvée dans un répertoire nommé `out`, qu'oXygen créée dans le répertoire courant si nécessaire.

Bien sûr, vous pouvez vérifier que le schéma généré depuis cet ODD valide correctement les documents de votre corpus. Ce serait alarmant si cela n'était pas le cas ! Mais lorsque vous aurez de nouveaux fichiers à ajouter à votre corpus, cependant, cette procédure peut devenir utile en fonction du fait que vous décidiez de maintenir des pratiques d'encodage déjà établies, ou bien pour les étendre afin de tenir compte des nouveaux usages dans vos fichiers. Vous aurez peut-être besoin d'ajouter de nouvelles valeurs parmi celles permises pour l'un de vos attributs. Peut-être devrez-vous réintroduire un nouvel attribut ou un élément dont vous pensiez ne jamais avoir besoin ?

Supposons par exemple que votre ODD indique que les valeurs de *@type* si elles sont indiquées sur `<div>` devrait être 'chapter' ou 'section', mais vos nouveaux matériaux incluent des éléments `<div>` avec un attribut *@type* avec la valeur 'book'. Vous pourriez décider que c'est une erreur et modifier le fichier, ou bien vous pourriez décider que vous avez besoin de modifier le fichier ODD. (Michael avait toujours l'habitude de dire 'S'il y a un manque de compatibilité entre le texte et le schéma, dans notre monde nous suivons le texte.') Disons-donc que vous adoptiez la dernière solution. Vous auriez alors besoin de localiser l'élément `<valList>` au sein de l'`<attDef>` concerné, et d'ajouter un nouvel élément `<valItem>`. De sorte que votre ODD ressemblerait alors à ça :

```
<elementSpec ident="div" mode="change">
 <attList>
  <attDef ident="type" mode="change">
   <valList mode="add" type="closed">
    <valItem ident="chapter"/>
    <valItem ident="section"/>
   </valList>
  </attDef>
 </attList>
</elementSpec>
```

Que vous changeriez pour ce qui suit :

```
<elementSpec ident="div" mode="change">
 <attList>
  <attDef ident="type" mode="change">
   <valList mode="add" type="closed">
    <valItem ident="chapter"/>
    <valItem ident="section"/>
    <valItem ident="book"/>
   </valList>
  </attDef>
 </attList>
</elementSpec>
```

Tant que vous êtes-là, vous pourriez aimer documenter ce que signifient chacune de ces valeurs, pour le bénéfice des non-anglophones, ou pour vous lorsque vous avez oublié si 'section' contient les 'book' ou l'inverse, ou bien vice-et-versa. Vous pouvez faire cela en ajoutant une ou deux descriptions supplémentaires en ajoutant les éléments `<desc>` au sein des différents `<valItem>`, comme ceci :

```
<elementSpec ident="div" mode="change">
 <attList>
  <attDef ident="type" mode="change">
   <valList mode="add" type="closed">
    <valItem ident="chapter">
     <desc xml:lang="en">the smallest subdivisions
           of a text</desc>
     <desc xml:lang="fr">les divisions les plus petites d'une
           texte</desc>
    </valItem>
    <valItem ident="section">
     <desc xml:lang="en">a group of
     <soCalled>chapter</soCalled>s within a
     <soCalled>book</soCalled>
     </desc>
     <desc xml:lang="fr">un regroupement de
     <soCalled>chapter</soCalled>s au sein d'un
     <soCalled>book</soCalled>
     </desc>
    </valItem>
    <valItem ident="book">
     <desc xml:lang="en">the largest subdivisions of
           a text</desc>
     <desc xml:lang="fr">les divisions les plus grandes d'une
           texte</desc>
    </valItem>
   </valList>
  </attDef>
 </attList>
</elementSpec>
```

Maintenant que vous avez fait ces changements, vous pourriez vouloir régénérer un schéma depuis votre ODD, puis pouvoir vérifier qu'oXygen utilise les nouvelles fonctionnalités que vous avez ajoutées. En ajoutant une `<div>` à un nouveau document, par exemple, vous devriez vous voir proposer un menu pour l'attribut *@type* incluant les nouvelles valeurs ainsi que l'aide qui leur est associée.

Maintenant, jeter un œil au 'mini-manuel' HTML généré depuis votre ODD. Par défaut, cela contiendra seulement un index des liens qui pointent vers des spécifications détaillées de chacun des éléments et des classes utilisées dans votre corpus. Ces descriptions sont apparemment modelée de près sur les spécifications employées dans les *Guidelines*.

Par défaut les exemples, les descriptions et les références fournies pour chaque élément seront les mêmes que celles des *Guidelines*. Le modèle de contenu et la liste des attributs pourraient toutefois refléter les changements que vous avez proposés dans votre ODD.

Vous pourriez vouloir tester cela en remplaçant un ou plusieurs des exemples d'utilisation fournis pour chaque éléments par des exemples tirés de votre propre corpus.

Ouvrir le fichier ODD que vous avez généré dans oXygen puis localiser l'élément `<elementSpec>` que vous voulez changer, s'il y en a un. (Sinon, vous devriez en ajouter un vous-même, bien évidemment.) Par exemple, supposons que je veuille fournir un meilleur exemple pour l'élément `<hi>` qui est inclus dans mon ODD. Une déclaration a été fournie pour cet élément qui efface les deux attributs globaux qui ne sont pas employés sur lui :

```
<elementSpec ident="hi" mode="change">
 <attList>
  <attDef ident="facs" mode="delete"/>
  <attDef ident="resp" mode="delete"/>
```

```
 </attList>
</elementSpec>
```

Mon nouvel exemple d'usage est fourni après <attList>. Il est encodé en utilisant l'élément <exemplum>, qui contient un élément <egXML> et (optionnellement) quelques paragraphes additionnels de discussion.

```
<elementSpec ident="hi" mode="change">
 <attList>
  <attDef ident="facs" mode="delete"/>
  <attDef ident="resp" mode="delete"/>
 </attList>
 <exemplum> <egXML xmlns="http://www.tei-c.org/ns/Examples"> Here's how I use
   it <hi>my example</hi> </egXML> </exemplum>
</elementSpec>
```

Plus de détails au sujet de l'élément <exemplum> et son contenu sont donnés dans la section dédiée des *Guidelines*, bien sûr. Notons que quand nous l'utilisons comme cela dans une personnalisation ODD, elle remplace complètement tous les <exemplum> existants dans la source TEI. Vous pouvez vouloir modifier votre ODD comme cela puis régénérer un mini-manuel HTML pour vérifier.

Avant cela, nous vous suggérons cependant d'aussi ajouter des explications au sujet de la destination de votre ODD, en documentant ses composants et leur emploi attendu. Cette documentation informelle peut être aussi simple ou complexe que vous le voulez, et vous pouvez (bien sûr) utiliser toute l'étendue des éléments fournis par la TEI pour l'exprimer. Ajouter au moins une ou deux, avec quelques éléments <head>, quelques <p> et quelques <list>… Vous pouvez aussi utiliser quelques balises de documentation plus spéciales fournies par le module tagdocs qui vous permettent de distinguer les noms d'attributs ou d'éléments par le balisage, et d'embarquer des descriptions formelles d'éléments spécifiques dans la prose, ou plein d'autres choses. Assurez-vous seulement que votre document ODD est valide contre un schéma qui inclue ces éléments spécialisés : le schéma tei_odds au sein du répertoire exemplars est destiné à cet usage.

Par exemple, vous pourriez étendre votre ODD pour arriver à quelque chose qui ressemblerait à ça :

```
<body>
 <div>
  <head>Un schéma minimal TEI pour la transcription d'archives</head>
  <p>Ce schéma propose un ensemble minimal d'éléments TEI adéquates pour une
     transcription élémentaire de sources archivistiques. </p>
  <p>Chaque document est transcrit avec un élément <gi>text</gi> distinct qui
     dipsose de son propre <gi>teiHeader</gi>. Les sous-sections du document
     sont marquées en utilisant l'élément <gi>div</gi>. </p>
  <p>Les éléments suivants de la TEI sont employés pour représenter ces
     composants : <specList>
    <specDesc key="TEI"/>
    <specDesc key="header"/>
    <specDesc key="text"/>
   </specList>
  </p>
<!-- lots more prose and discussion and examples here-->
 </div>
 <div>
  <head>Spécification formelle</head>
```

```
  <schemaSpec ident="minimalSchema"
   start="TEI">
<!-- declarations from your generated ODD here -->
  </schemaSpec>
 </div>
</body>
```

Lorsqu'un ODD comme ceci est traité par un processeur, la sortie HTML (ou les autres formats de document) générée par un processeur ODD contiendra l'ensemble du texte que vous voyez ici. L'élément `<specDesc>` sera toutefois remplacé par le contenu de l'élement `<desc>` au sein de la spécification de l'élément indiquée par la valeur de l'attribut *@key*. Le `<schemaSpec>` sera aussi traité ainsi que nous l'avons vu plus haut pour produire une documentation formelle. Essayez !

Il y a beaucoup d'autres choses que vous pourriez vouloir faire en commençant à éditer votre ODD. Ce tutoriel en suggère seulement quelques unes qui s'avèrent particulièrement utiles dans le processus de définition d'un schéma utile pour un ensemble de documents TEI.